6

BEST AVAILABLE COPY

UNITED STATES DEPARTMENT OF COMMERCE
Unit d Stat s Patent and Trad mark Offic
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
08/880,616	06/23/97	COHEN M	Y0997-111

TM02/0522

ANNE VACHON DOUGHERTY
3173 CEDAR ROAD
YORKTOWN HEIGHTS NY 10598

EXAMINER	
FOURSON, G	
ART UNIT	PAPER NUMBER
2151	16

DATE MAILED: 05/22/01

Please find below and/or attached an Office communication concerning this application or proceeding.

Commissioner of Patents and Trademarks

PTO-90C (Rev.11/00)

1- File Copy

Office Action Summary	Application No. 08/880,616	Applicant(s) Cohen et al.
	Examiner Gary Fourson	Art Unit 2151

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136 (a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1) ☒ Responsive to communication(s) filed on *Feb 12, 2001*.

2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.

3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11; 453 O.G. 213.

Disposition of Claims

4) ☒ Claim(s) *1-14 and 16-20* is/are pending in the application.

4a) Of the above, claim(s) ______ is/are withdrawn from consideration.

5) ☐ Claim(s) ______ is/are allowed.

6) ☒ Claim(s) *1-14 and 16-20* is/are rejected.

7) ☐ Claim(s) ______ is/are objected to.

8) ☐ Claims ______ are subject to restriction and/or election requirement.

Application Papers

9) ☐ The specification is objected to by the Examiner.

10) ☐ The drawing(s) filed on ______ is/are objected to by the Examiner.

11) ☐ The proposed drawing correction filed on ______ is: a) ☐ approved b) ☐ disapproved.

12) ☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

13) ☐ Acknowledgement is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d).

a) ☐ All b) ☐ Some* c) ☐ None of:

1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. ______.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

*See the attached detailed Office action for a list of the certified copies not received.

14) ☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

15) ☒ Notice of References Cited (PTO-892)

16) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

17) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s). ______

18) ☐ Interview Summary (PTO-413) Paper No(s). ______

19) ☐ Notice of Informal Patent Application (PTO-152)

20) ☐ Other:

DETAILED ACTION

This final rejection is responsive to Amendment D (paper no. 15), received February 12, 2001.

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. **Claims 1 and 11 are rejected under 35 U.S.C. 103(a) as being unpatentable over Zolnowsky (5,826,081) in view of Custer (Inside Windows NT) and further in view of Admitted Prior Art (Applicant's Specification page 2 line 20 through page 3 line 7).**

With respect to claims 1 and 11, Zolnowsky teaches a plurality of tasks (threads) of more than one application (Single processor, multitasking operating systems such as Windows NT described by Custer, see page 83 line 3, routinely schedule threads from more than one application. Adding a multiplicity of processors for greater computing throughput for multiple applications was and has been a highly desirable feature in the multitasking art. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to execute a plurality of tasks from more than one application on the multiprocessor system, because the real time applications of Zolnowsky are typically partitionable into schedulable entities or mixed-mode applications, see column 2 lines 35-40.), computing nodes (col. 3 line 52, "multiple processors P1 through Pn"), a plurality of local processes (Zolnowsky defines tasks as the

unit of resource management and the thread as a single flow-of-control in column 1 lines 31-34. In column 6 lines 63-65 indicate that some threads are only executable on a particular processor of the multiprocessor assembly. Therefore, there is a process [or task as Zolnowsky has so defined] that is different on that particular processor from other processes on other processors.), providing application information to scheduler means (column 5 lines 26-38), and dynamically creating a prioritized schedule of a plurality of tasks of the more than one application (col. 6 line 45, "Also, when a thread is made runnable, it is placed on a dispatch queue, ..."; col 8 line 66, "Real time threads are scheduled strictly on the basis of their priority ...") to allow execution of tasks of more than one application at the same time (Single processor, multitasking operating systems such as Windows NT described by Custer, see page 83 line 3, routinely schedule threads from more than one application.), a local scheduler (column 7 line 15, "Each processor ..."), and means for ascertaining [determining] which process(es) are assigned [correspondence] to the tasks (In column 6 lines 63-65 indicate that some threads are only executable on a particular processor of the multiprocessor assembly. Therefore, there is a process on that particular processor that is different from other processes on any other processor. Custer teaches on page 84 that a process must be combined with a thread of execution before it can do any work. The thread of execution "is the entity within a process that the NT kernel schedules for execution. Without it, the process's program cannot run." Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate means for ascertaining which process(es) are assigned to the tasks, because Custer recognized without the correlation between processes and threads the process or program would not be able to execute.). However,

Zolnowsky as modified by Custer does not appear to teach means for prioritizing the local processes according to the prioritized schedule.

Applicant has divulged on pages 2-3 that the AIX™ operating system assigns a common priority to the process(es) required for (or correlated to) a task. Having the processes, associated with individual tasks, assigned priorities corresponding to the schedule would have been a highly desirable feature in the art. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the local scheduling/correlating means of IBM into the task scheduling system of Zolnowsky, because prioritizing local processes with the current and next task would have been expected to result in higher cache preloading efficiency.

3. **Claims 1, 11, 12, and 14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Boland et al. (5,872,972) in view of Custer (Inside Windows NT) and further in view of Admitted Prior Art (Applicant's Specification page 2 line 20 through page 3 line 7).**

With respect to claims 1 and 11, Boland et al. teaches a plurality of tasks (processes) of more than one application (col 4 line 19, "all runnable processes in global run queue 24, ..."), computing nodes (col 3 line 52), a plurality of local processes (col 4 line 21, "processes which have been previously run and are now affinitized to a specific processor." Col 4 line 59, "once a processor runs a process, it would never age away its affinity from that processor."), providing application information to global scheduler means (The scheduler 90 obviously takes information from applications or processes to schedule the processes into the global priority queue. In column 3 lines 1-5 processes provide affinity information to the scheduler.), and dynamically creating a schedule of a plurality of tasks utilizing priorities (col 4 lines 22-24), and a local scheduler comprising

means for receiving said global prioritized schedule (Column 4 lines 26-31 teach that each 'processor' consults the global queue.) as well as means to update a local priority list to include said assigned processes (Referring to Figure 7, column 7 lines 26-28, "These processes may thereafter be reordered based upon process priority within a nodal priority run queues 71 and 77 ..."). However, Boland does not explicitly refer to the runnable processes in column 4 lines 16-25 as being from a plurality of applications.

Custer teaches that single processors with multitasking operating systems routinely execute process threads from more than one application. [page 83 line 3] Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to actively schedule processes from more than one application on a multi-processor system.), , means for ascertaining which process(es) are assigned to the tasks (Custer teaches on page 84 that a process must be combined with a thread of execution before it can do any work. The thread of execution "is the entity within a process that the NT kernel schedules for execution. Without it, the process's program cannot run." Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate means for correlating or ascertaining which process(es) are assigned to the tasks, because Custer recognized without the correlation between processes and threads the process or program would not be able to execute.).

As to means for prioritizing the processes according to the prioritized schedule (Applicant has divulged on pages 2-3 that the AIX™ operating system assigns a common priority to the process(es) required for (or correlated to) a task. Having the processes, associated with individual tasks, assigned priorities corresponding to the priorities of the schedule would have been a highly desirable feature in the art. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to

incorporate the local scheduling/correlating means of IBM into the task scheduling system of Cameron et al., because prioritizing local processes according to the task correlation would have been expected to result in higher cache preloading efficiency.).

As to **claims 12 and 14**, invoking operating system priorities to schedule tasks in accordance with said prioritized schedule: The operating system would inherently follow any prioritizing scheme employed by the programmer or else there would not be any need to incorporate the local/global scheduling means in the first place.

4. **Claims 2, 4-10, 13-16, and 18-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Boland et al. (5,872,972) in view of Custer (Inside Windows NT), Admitted Prior Art (Applicant's Specification page 2 line 20 through page 3 line 7), and further in view of Cameron et al. (5,325,526).**

As to **claim 2**, said computing node comprising an operating system for "receiving input" from the prioritizing means and "directing said assigned processes" to execute tasks in a prioritized order: Cameron et al. in Figures 4 and 5 show a prior art task scheduler. Column 5 last paragraph elaborates stating that each scheduler comprises operating system software responsible for controlling the execution of a plurality of tasks. It would have been obvious to one ordinarily skilled in the art at the time the invention was made for the OS to receive information about the execution of the plurality of tasks as taught by Cameron et al. with the task scheduling system of Boland et al., because Cameron et al. states in column 6 lines 28-31, "Interactive scheduling using Unix, or other operating systems in a single processor environment, is well known to those of ordinary skill in the art."

As to **claim 4**, application coordinator means for communicating information to said scheduler: Scheduling information must inherently be obtained by some means in

order to produce a prioritized list of tasks, however in column 8, Cameron et al. teaches on line 2, "The allocator and scheduler 710 comprises processing logic and data for allocating nodes to specific application programs and for scheduling applications programs for execution." The "Make Partition" procedure (720) is the request for the allocator/scheduler to initialize tasks which as stated on line 18, "are retrieved and loaded into the nodes associated with the specified partition."

As to **claim 5**, said local processes being adapted to perform tasks in parallel: Also, in column 2 on line 50 Cameron teaches that application programs are allowed to execute on one or more nodes of a partition. Furthermore, column 7 line 40 states, "...an entire application program is active at once across all of the nodes on which the application program is loaded." The multi-node or multi-processor collaborative effort to the processing of a set of tasks or application program processes is the truest definition of parallel processing. Cameron et al. in column 1 on lines 26 to 30 indicates that multi-tasking, round robin processing, time slicing, or parallel processing was well known to one of ordinary skill in the art at the time the invention was made.

As to **claim 6**, said scheduler means comprising global scheduler means which in turn comprises means for dynamically scheduling then communicating the schedule to the local scheduler: Cameron et al. teaches the local nodes are assigned to application programs. The allocator and scheduler 612 act functionally as a "global scheduler" by controlling and assigning the appropriate nodes from a particular layer. Column 7 line 50 states, "As will be described below, allocator and scheduler 612 may and typically does operate with a plurality of partitions 614." In column 9 on line 50, "In the preferred embodiment, partition data blocks and application data blocks can be maintained in the same doubly-linked list." Further down on line 64 it is stated that, "The current priority

field 918 may dynamically change as the priorities of associated application programs or sub-partitions change priority." Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize means for dynamically scheduling then communicating the schedule to a local scheduler as taught by Cameron et al. with the task scheduling means of Boland, because Cameron et al. recognized, "The current priority field 918 may dynamically change as the priorities of associated application programs or sub-partitions change priority." [line 64]

As to **claim 7**, said local scheduler being adapted to communicate process information to the global scheduler: Cameron et al. teaches in column 14 lines 12-31 three access modes to the partition data. They are read, write, and execute access modes allowing or disallowing the ability to run application programs from a partition and to create or remove sub-partitions from a partition. This information is also available to the allocator/scheduler 710. Also, figure 7 shows application data 736 specifically available to the allocator/scheduler.

As to **claim 8**, the global scheduler also comprising timer means to effect schedule communication: Cameron et al. teaches in column 11 lines 6-11 a time executed field 1021.

As to **claim 9**, said global scheduler including a local scheduler address table: Cameron et al. teaches in column 13 lines 15-33, "Two hash tables providing a quick look-up mechanism for locating partitions ..."

As to **claim 10**, Boland et al. as modified by Custer (Inside Windows NT), Admitted Prior Art (Applicant's Specification page 2 line 20 through page 3 line 7), and as further modified by Cameron et al. for the rejection of claims 1, 2, and 6 teaches the limitations as claimed.

As to **claim 19**, the communicating and updating steps noted above as taught by Boland do not require user input, and, therefore, are automatically performed.

As to **claim 13**, scheduler means is remote to the node and communicating the schedule to the node: Cameron et al. shows in Figures 4 and 5 that in prior art methods of task management systems, the Scheduler 410, 510 can be remotely located from the processors. In column 6 lines 32-45 refer specifically to Figure 5 noting that the scheduler arranges an orderly schedule for multiple tasks executing on multiple processors. Line 37 mentions a common memory where the schedule information would be communicated to the three processing nodes. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have scheduler means of Boland et al. as modified remote to the node as taught by Cameron et al., because Cameron et al. recognized mulitprocessor systems require global scheduling means remote from the majority of computing nodes.

As to **claim 15**, communicating task execution information to the scheduler: Cameron et al. teaches in column 11 lines 6-11 a time executed field 1021. This "execution information" is part of a process group field 1020 which is updated and available to the scheduler.

As to **claim 16**, repeating said steps until all tasks have been completed: Cameron teaches recursive scheduling in column 15 on lines 12-14.

As to **claim 18**, said remotely located scheduler dynamically maintaining a computing node's list: Figure 7, Layer Data -738-; Column 9 lines 28-31, "The layer data structure 738 comprises information including identity of the nodes of the partition that are allocated by a list of consumers to which the layer points."

As to **claim 20**, Boland teaches receiving task information from an application coordinator [scheduler 90] and maintaining an activity scheduler list [The nodal run queues comprise processes previously run on a particular processor for which the data and instruction caches may have an affinity. See col. 4 lines 37-39] and an activity priority list [priority queues].

5. **Claims 3 and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Boland et al. (5,872,972) as modified by Custer (Inside Windows NT), Admitted Prior Art (Applicant's Specification page 2 line 20 through page 3 line 7), and as further modified by Cameron et al. (5,325,526) as applied to claims 2 and 14 above, and further in view of Ripps (The Multitasking Mindset Meets the Operating System).**

As to **claim 3**, the operating system being further adapted to interleave local operations with said tasks: A node or CPU controlled by an operating system would inherently process local operations (e.g. an exception) pertaining to the operating system commands. Ripps teaches on page 9 that C and proprietary OS functions are intermixed in a typical task. Context switches controlled by the operating system are also well known local tasks which are interleaved between the application task execution.

As to **claim 17**, Boland et al. as modified by Custer (Inside Windows NT), Admitted Prior Art (Applicant's Specification page 2 line 20 through page 3 line 7), Cameron et al., and as further modified by Ripps for the rejection of claims 1, 3, 5, 11, 12, and 14 teaches the limitations substantially as claimed.

Response to Amendment

6. **Applicant's arguments filed February 12, 2001 have been fully considered, but they are not persuasive.**

7. Applicant has presented the following arguments noting apparent deficiencies in the rejections of claims 1-18 under U.S.C. 103(a) as being unpatentable over Zolnowsky in view of Custer and further in view of Admitted Prior Art:

a. Dynamically creating a global prioritized schedule of a plurality of tasks of more than one application: Applicant states on page -8-, "The Zolnowsky patent provides a dispatcher model which maintains a global dispatch queue for non-bound higher priority real time threads." The requirement of a "global prioritized schedule" does not require all tasks in the multiprocessor system to be included in the list but rather that it contain a plurality of tasks of more than one application. Figure 2(B) illustrates a Superqueue as well as individual dispatch queues for tasks assigned to a particular processor.

b. Communicating the global prioritized schedule to more than one computing node (at which the local prioritized schedule is updated-Not required by the limitations of Claim 11): Claim 1 recites 'means for receiving the global prioritized schedule,' and Applicant states on page -8-, "Each processor in the multiprocessor environment selects a candidate thread from its own queue, compares the selected thread with threads in the global real-time queue, ..." This comparison process reads on the limitation of 'communicating said global prioritized schedule to said more than one computing node' as required by Claim 11 and, as broadly interpreted, 'means for receiving said global prioritized schedule' in claim 1.

c. Determining correspondence between the plurality of tasks and a plurality of local processes: As noted by Custer threads are inherently associated with a process due to the fact that threads are the execution mechanism of processes. Without a thread the process would not execute anything and threads do not exist without a process space within which to execute.

d. At least one local scheduler associated with each of the more than one computing nodes, *which each have a plurality of processes*: Examiner notes that local scheduling is not required to be accomplished by a software process distinct from the global scheduling algorithms.

e. Dynamically prioritizing the local processes in accordance with the global prioritized schedule allowing simultaneous execution of tasks from more than one application: As to allowing simultaneous execution of tasks from more than one application, the limitation does not require tasks from more than one application to execute on any one processor at any moment. The system of Cameron et al. teaches dividing a multiprocessor complex into partitions where one application is active on any one partition at a time. However, the plurality of processors may be divided into multiple partitions where at any given instance multiple partitions may execute tasks from multiple applications concurrently as claimed. Furthermore, prioritized execution of processes and, therefore, threads is not new. As was known by those of ordinary skill in the art at the time the invention was made, multiple queues (linked lists of scheduled tasks) for each scheduler may exist each of which correspond to a particular priority level.

f. Association of a task's priority with the corresponding process priority: Custer teaches on page 84 that a process must be combined with a thread of execution before it can do any work. The thread of execution "is the entity within a process that the NT kernel schedules for execution. Without it, the process's program cannot run."

8. Applicant has presented the following arguments noting apparent deficiencies in the rejections of claims 1-18 under U.S.C. 103(a) as being unpatentable over Boland et al. in view of Custer and further in view of Admitted Prior Art: Applicant

argues on page 12 concluding from noted teachings of Boland et al. "that there is [a] no multinode environment, [b] no local scheduler, [c] no local scheduler prioritized list, [d] no communication of a global prioritized list to local nodes, and [e] no updating of a local prioritized list based on the global prioritized list." [lettering added] Applicant also concludes or summarizes on page -13- "Boland does not [f] create a global prioritized list, does not [c] maintain or [e] update a local prioritized list, and does not actively [d] communicate any priority information (i.e., a global prioritized list) to the local nodes for updating a local prioritized list." [lettering added]

a. Applicant states that Boland relates to scheduling or work distribution "in a multiprocessor environment having one node..." Examiner notes a contradiction between the multiple processor system (multiprocessor environment: Figures 1, 2, 6, and 7) and the conclusion "there is no multinode environment."

b. Examiner notes as above that local scheduling is not required to be accomplished by a software process distinct from the global scheduling algorithms.

c. Column 7 in reference to Figure 7 elaborates and illustrates as (Applicant has noted) that an alternative embodiment includes process scheduling (run) queues for both global, non-affined processes as well as local, affined processes.

d. As to communicating priority information to local nodes, Boland teaches that available processors check or consult [col. 7 lines 30-34] the global priority run queue comparing priorities of processes on the global and local run queues to determine the process with the highest priority. This consultation, as broadly interpreted, teaches communicating priorities to the local nodes' schedulers.

e. Examiner notes that the limitation pertaining to updating a local prioritized list is not recited in independent claim 11. In column 7 lines 19-30 Boland notes that the nodal priority run queues are updated with processes that have previously run on that particular node and that the processes may be reordered thereafter based upon process priority within the queues.

f. See Boland, column 4 lines 16-25: The global run queue is a prioritized ordering of non-affined tasks and is accessible by local scheduling threads as noted in section 8d. APA teaches that the AIX™ operating system assigns a common priority to the process(es) required for (or correlated to) a task. It would have been obvious to one of ordinary skill in the art at the time the invention was made to associate priorities of process(es) and related threads, because Custer recognized without the correlation between processes and threads the process or program would not be able to execute.

Pertinent Prior Art

9. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure: Freedman et al. (US 4,318,173), Farrell et al. (US 5,630,128), and Jagannathan et al. (US 5,692,193) teach coordinated multiprocessor scheduling.

Conclusion

10. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any

extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication should be directed to **Gary Fourson** at telephone number **(703) 305-4392** or E-mail at the address **gary.fourson@uspto.gov**.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

The fax numbers for formal (703-308-9051), to be intended for entry into the application, or informal (703-305-9731) communications may be utilized for expedited transactions.

gsf

May 17, 2001

ALVIN OBERLEY
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100